...e costruisci il tuo **LABORATORIO DIGITALE**

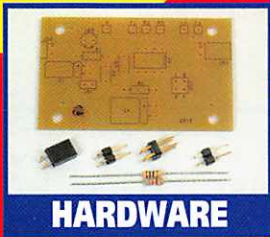
6,90 €

53

HARDWARE

DIGITALE DI BASE

MICROCONTROLLER

DIGITALE AVANZATO

TOTALMENTE PROGRAMMABILE!!!

9 771824 363008 41053

Peruzzo & C.

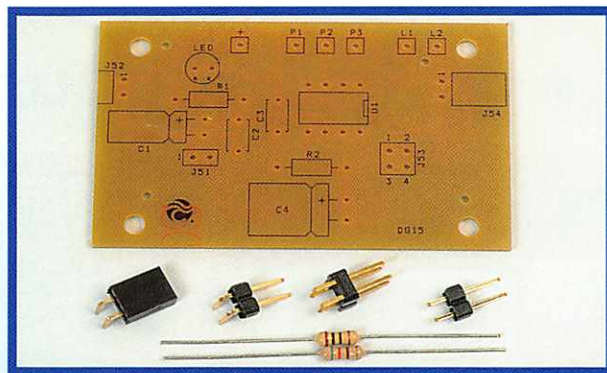
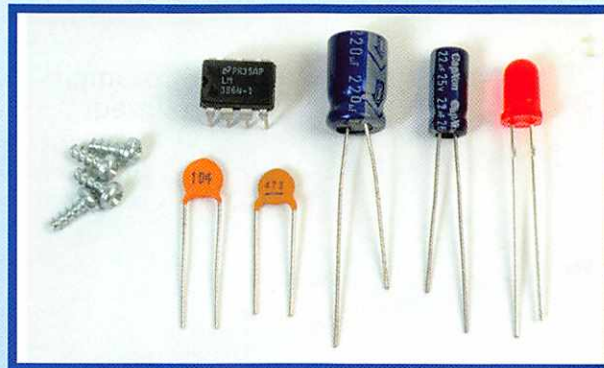
Direttore responsabile:
ALBERTO PERUZZO
Direttore Grandi Opere:
GIORGIO VERCELLINI
Consulenza tecnica e traduzioni:
CONSULCOMP S.n.c.
Pianificazione tecnica
LEONARDO PITTON

Direzione, Redazione, Amministrazione: viale Ercole Marelli 165, Tel. 02/242021, 20099 Sesto San Giovanni (Mi). Pubblicazione settimanale. Registrazione del Tribunale di Monza n. 1738 del 26/05/2004. Spedizione in abbonamento postale gr. II/70; autorizzazione delle Poste di Milano n. 163464 del 13/2/1963. Stampa: Grafiche Porpora s.r.l., Cernusco S/N (MI). Distribuzione SO.DI.P. S.p.A., Cinisello Balsamo (MI).

"ELETTRONICA DIGITALE" si compone di 70 fascicoli settimanali da suddividere in 2 raccoglitori.

RICHIESTA DI NUMERI ARRETRATI. Per ulteriori informazioni, telefonare dal lunedì al venerdì ore 9.30-12.30 all'ufficio arretrati tel. 02/242021. Se vi mancano dei fascicoli o dei raccoglitori per completare l'opera, e non li trovate presso il vostro edicolante, potrete riceverli a domicilio rivolgendovi direttamente alla casa editrice. Basterà compilare e spedire un bollettino di conto corrente postale a PERUZZO & C. s.r.l., Ufficio Arretrati, viale Marelli 165, 20099 Sesto San Giovanni (MI). Il nostro numero di c/c postale è 42980201. L'importo da versare sarà pari al prezzo dei fascicoli o dei raccoglitori richiesti, più le spese di spedizione € 3,10 per pacco. Qualora il numero dei fascicoli o dei raccoglitori sia tale da superare il prezzo globale di € 25,82 e non superiore a € 51,65, l'invio avverrà per pacco assicurato e le spese di spedizione ammonteranno a € 6,20. La spesa sarà di € 9,81 da € 51,65 a € 103,29; di € 12,39 da € 103,29 a € 154,94; di € 14,98 da € 154,94 a € 206,58; di € 16,53 da € 206,58 in su. Attenzione: ai fascicoli arretrati, trascorse dodici settimane dalla loro distribuzione in edicola, viene applicato un sovrapprezzo di € 0,52, che andrà pertanto aggiunto all'importo da pagare. Non vengono effettuate spedizioni contrassegno. Gli arretrati di fascicoli e raccoglitori saranno disponibili per un anno dal completamento dell'opera. IMPORTANTE: è assolutamente necessario specificare sul bollettino di c/c postale, nello spazio riservato alla causale del versamento, il titolo dell'opera nonché il numero dei fascicoli e dei raccoglitori che volete ricevere.

IN REGALO in questo fascicolo

- 1 Scheda DG15
- 1 Resist. 10 Ohm 5% 1/4 W
- 1 Resist. 820 Ohm 5% 1/4 W
- 1 Connettore femmina da c.s. a 90° a 2 vie
- 1 Connettore maschio da c.s. a 90° a 2 vie
- 1 Connettore maschio da c.s. diritto a 2 vie
- 1 Connettore maschio da c.s. diritto a 2 file x 2 vie

IN REGALO nel prossimo fascicolo

- 1 Integrato LM386
- 1 Condensatore elettrolitico 220 µF
- 1 Condensatore elettrolitico 22 µF
- 1 Condensatore ceramico 47 nF
- 1 Condensatore ceramico 100 nF
- 1 LED rosso
- 4 Viti

COME RACCOGLIERE E SUDDIVIDERE L'OPERA NELLE 4 SEZIONI

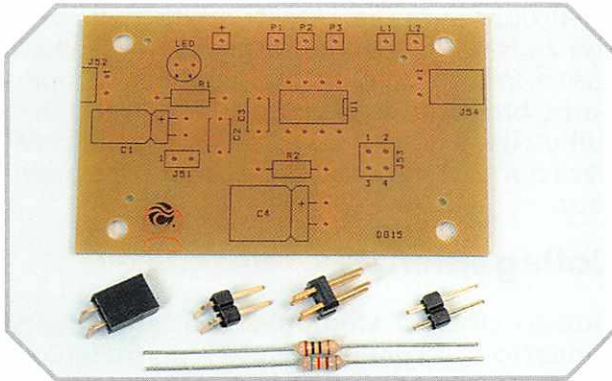
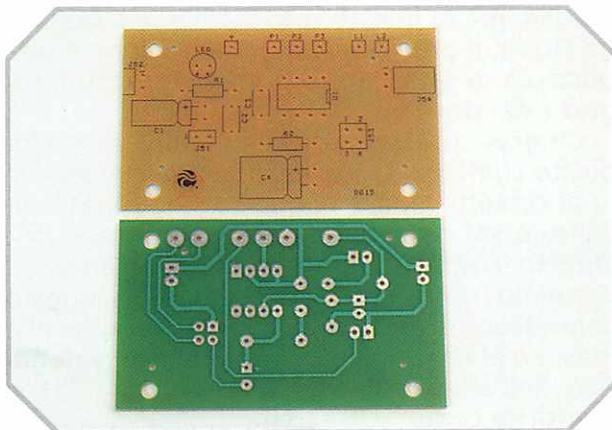
L'Opera è composta da 4 sezioni identificabili dalle fasce colorate, come indicato sotto. Le schede di ciascun fascicolo andranno suddivise nelle sezioni indicate e raccolte nell'apposito raccoglitore, che troverai presto in edicola. Per il momento, ti consigliamo di suddividere le sezioni in altrettante cartellette, in attesa di poterle collocare nel raccoglitore. A prima vista, alcuni numeri di pagina ti potranno sembrare ripetuti o sbagliati. Non è così: ciascuno fa parte di sezioni differenti e rispecchia l'ordine secondo cui raccogliere le schede. **Per eventuali domande di tipo tecnico scrivere al seguente indirizzo e-mail:** elettronicadigitale@microrobots.it

Hardware Montaggio e prove del laboratorio

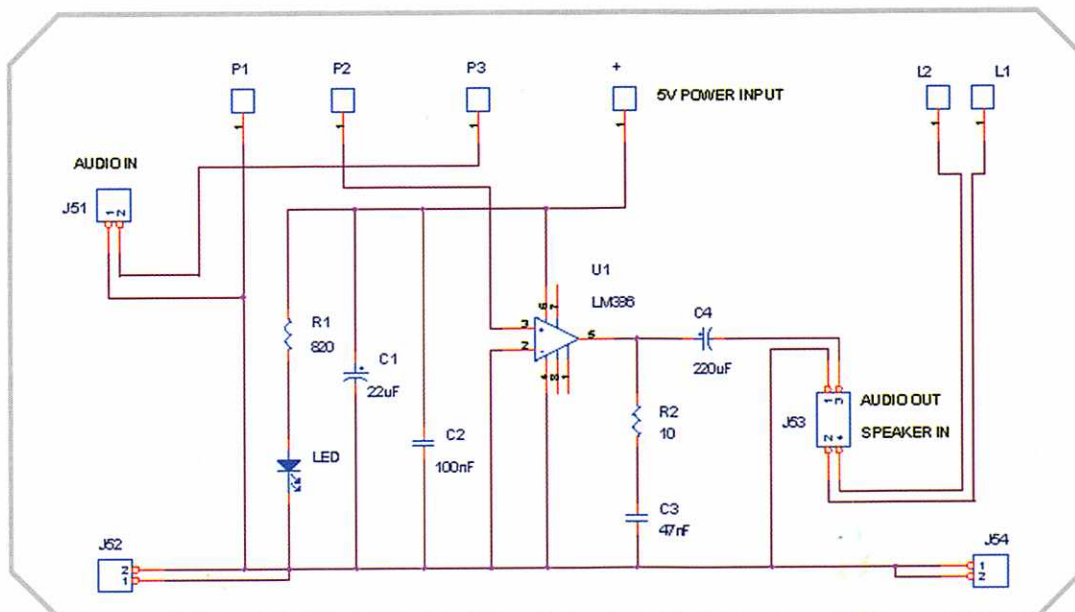
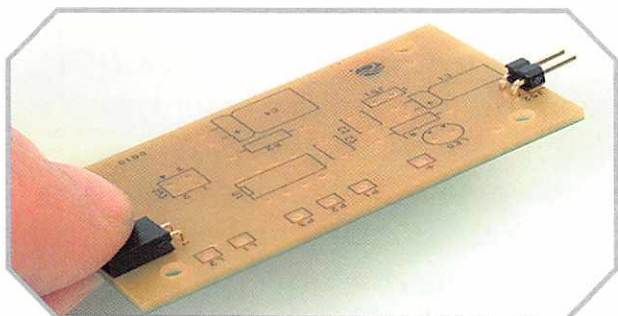
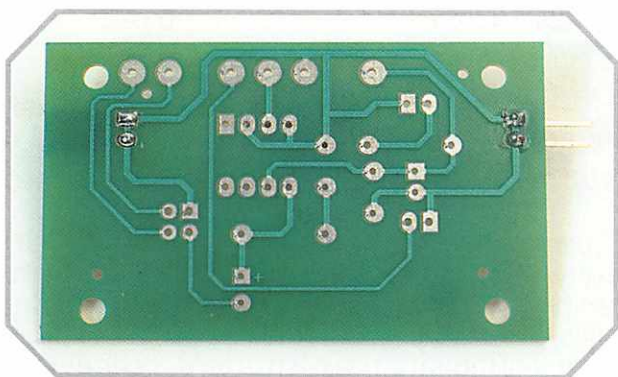
Digitale di base Esercizi con i circuiti digitali

Digitale avanzato Esercizi con i circuiti sequenziali

Microcontroller Esercizi con i microcontroller

L'amplificatore audio

Componenti allegati a questo fascicolo.

Circuito stampato DG15.

Con questo fascicolo vengono forniti il circuito stampato DG15, i connettori da inserire nello stesso e due resistenze. Inizieremo il montaggio dei componenti della scheda DG15, che è un amplificatore audio, elemento ausiliario molto importante nel laboratorio, in quanto riceve segnali elettrici ed emette segnali audio, se le frequenze ricevute sono all'interno della banda audio. Si utilizza per evidenziare la presenza di segnali la cui frequenza è compresa nello spettro audio, rendendo così superflua la strumentazione.

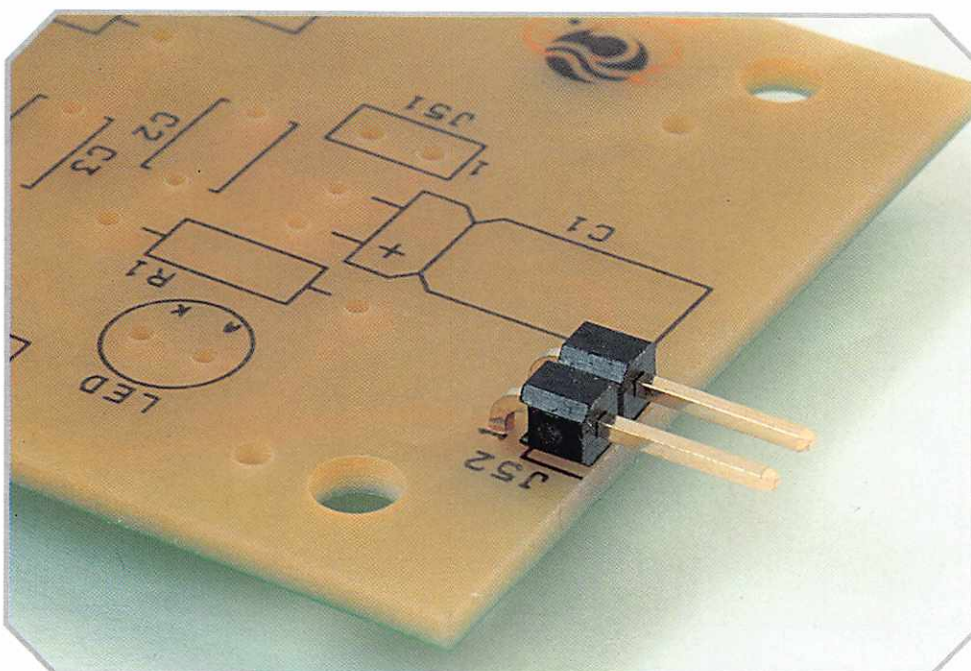
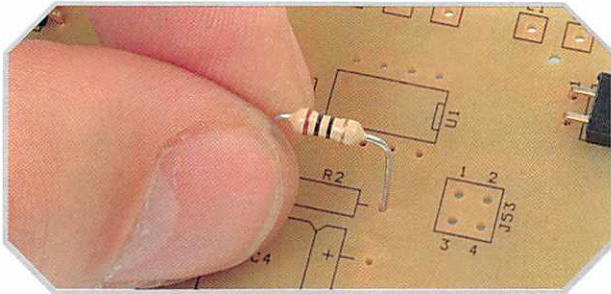
Questo amplificatore audio è basato sul circuito integrato LM386, con il quale si ottiene un funzionamento molto stabile e si evita l'utilizzo di elementi di regolazione. Si usa il circuito consigliato dal costruttore, i condensatori C1 e C2 filtrano l'alimentazione, il condensatore C4 svolge la funzione di disaccoppiare l'uscita che è tipica di questo tipo di amplificatori, evita il passaggio di corrente continua sull'uscita, impedendo che questa arrivi

Schema elettrico dell'amplificatore audio.

Inserimento dei connettori del negativo dell'alimentazione.

Scheda capovolta con i connettori saldati.

all'altoparlante. La rete formata dalla resistenza R2 e dal condensatore C3 evita che il carico induttivo, rappresentato dall'altoparlante, provochi oscillazioni ad alta frequenza sull'uscita. Il diodo LED e la resistenza di polarizzazione R1 indicano che il circuito è alimentato.

Collegamenti

Questo circuito stampato contiene diversi connettori. Il connettore J52 tipo maschio a 90° si utilizza per il collegamento del negativo dell'alimentazione alla scheda DG16. Questa stessa funzione di collegamento del negativo la realizza anche il connettore femmina a 90° J54, ma in questo caso si collega alla scheda DG14. Il positivo dell'alimentazione è una piazzola di saldatura e si identifica con il segno + del positivo.

L'ingresso del segnale all'amplificatore è indicato come P2, tuttavia l'ingresso del segnale al circuito, cioè il segnale audio a basso livello, è sul terminale 2 del connettore J51. Questo connettore è accessibile tramite il pannello frontale e l'altro terminale di questo connettore – terminale 1 – corrisponde alla massa o al riferimento ed è collegato al negativo dell'alimentazione. Il connettore J51 si identifica come AUDIO IN.

Osservando lo schema vedremo che è ne-

I connettori devono rimanere ben allineati.

La resistenza R1 è da 820 Ω e la R2 solamente da 10 Ω.

Resistenze montate.

cessario passare tutto o parte del segnale del terminale P3 al P2, questo lo faremo con un potenziometro – quello del volume – in seguito vedremo i dettagli del suo funzionamento e dei collegamenti.

L'uscita del circuito è sul terminale 3 del connettore J53, il quale può essere collegato all'altoparlante tramite un ponticello, sui terminali 3 e 4 di questo connettore. L'altoparlante a sua volta si collega alle piazzole di saldatura L1 e L2 di questo stesso circuito.

I terminali 1 e 2 di questo connettore corrispondono ai terminali di massa e sono collegati al negativo dell'alimentazione. Per collegare l'altoparlante del laboratorio è necessario collegare fra loro, con un altro ponticello, anche i terminali 1 e 2.

I quattro terminali di questo connettore sono accessibili dal pannello frontale, i terminali 1 e 3 corrispondono all'uscita audio e sono etichettati come AUDIO OUT. Il terminale di uscita è indicato con un punto rosso e quello della massa con uno nero. I terminali 2 e 4 corrispondono ai collegamenti dell'altoparlante che si identifica come SPEAKER OUT.

L'utilizzo di questo connettore permette l'uso indipendente dell'amplificatore e dell'altoparlante.

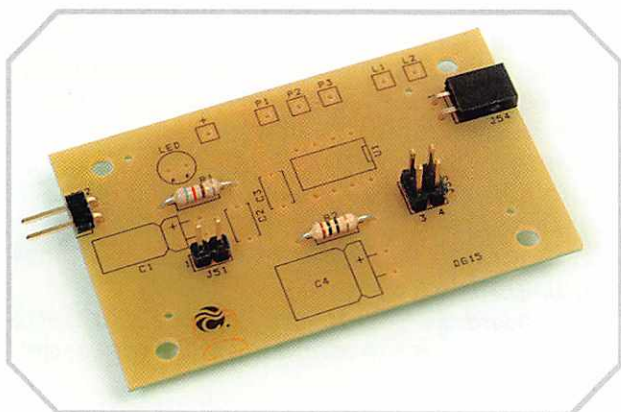
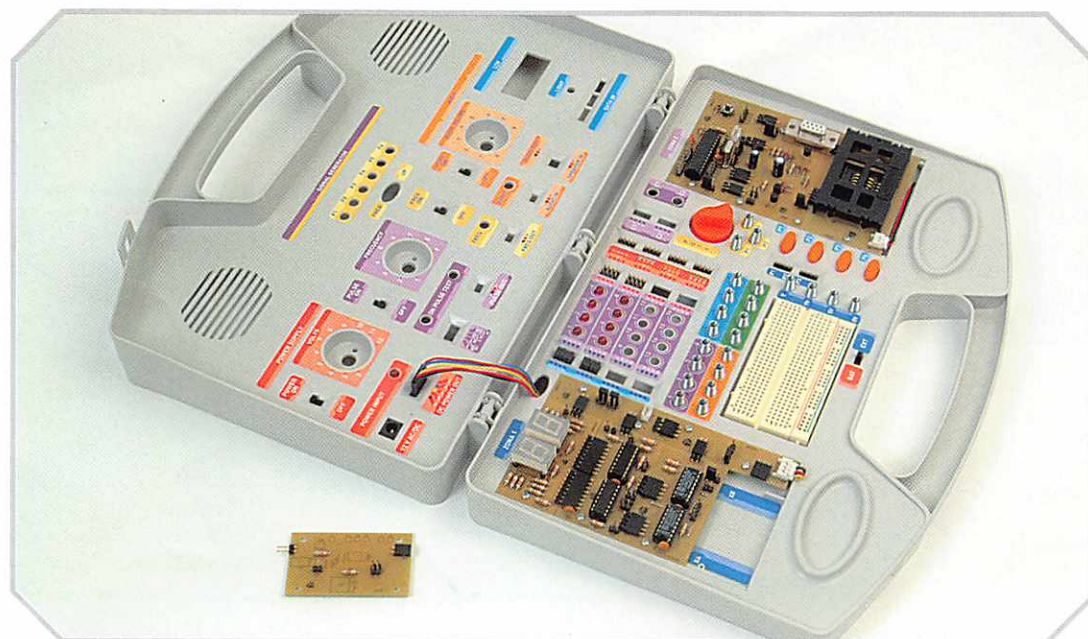
Montaggio dei connettori J51 e J53.

Montaggio

I primi componenti che si devono montare sono i connettori J52 e J54. J52 è maschio e J54 è femmina, entrambi sono del tipo a 90°, inoltre è facile identificare la loro posizione grazie alla serigrafia e aiutandosi con le fotografie. Devono rimanere ben appoggiati e allineati alla scheda, per poter realizzare un buon collegamento con le schede situate a sinistra e a destra della stessa.

Dopo inseriremo le due resistenze, tenendo presente che R1 è da 820 Ω (grigio, rosso, marrone) e R2 è da 10 Ω (marrone, nero, nero). Piegheremo i terminali e li inseriremo nei fori dove vengono montate le resistenze e le salderemo. Taglieremo poi la parte in eccesso dei reofori.

I connettori J51 a due vie e J53 a quattro sono del tipo maschio diritto. Dopo averli inseriti nella scheda, la capovolgeremo appoggiandola sui connettori stessi e salderemo solamente un terminale per ognuno dei connettori, verificando che siano perfettamente verticali. Nel caso fossero posizionati male è possibile correggere la posizione, riscaldando e fondendo nuovamente lo stagno del terminale saldato. Quando la posizione sarà corretta salderemo il resto dei terminali. Per il momento non abbiamo altro materiale da montare.

Saldatura dei connettori J51 e J53.

Circuito stampato così come deve risultare.

Vista generale del laboratorio e della scheda DG15 allo stato attuale del montaggio.

Circuito combinazionale

Un'applicazione tipica è l'implementazione di funzioni per le quali l'uscita o le uscite dei circuiti si attivano solamente con determinate combinazioni applicate agli ingressi.

Il problema

Abbiamo a disposizione tre variabili di ingresso e una di uscita. Le tre variabili sono rappresentate da tre pulsanti P1, P2 e P3 e il livello dell'uscita si visualizza con un LED, il quale si illumina quando questa uscita è a livello alto e si spegne quando è a livello basso.

Per quanto riguarda gli ingressi, il livello uno corrisponde al pulsante premuto, il livello zero al pulsante in riposo.

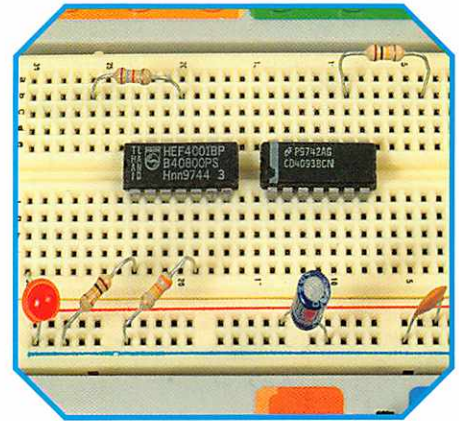
Vogliamo che l'uscita sia a livello alto solamente con le seguenti combinazioni di pulsanti:

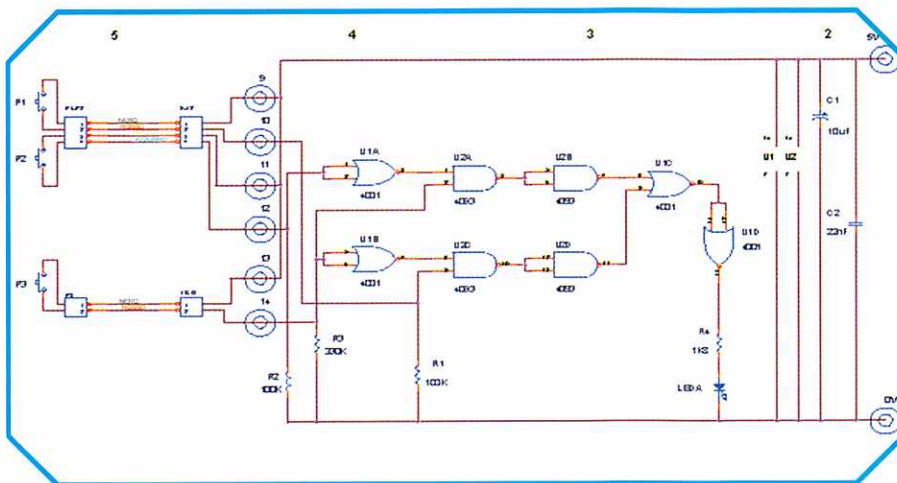
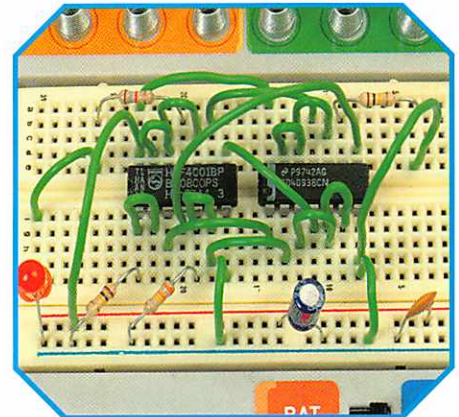
- P3 attivato, P2 in riposo, P1 in riposo.
- P3 in riposo, P2 attivato, P1 attivato.
- P3 in riposo, P2 in riposo, P1 attivato.
- P3 attivato, P2 in riposo, P1 attivato.

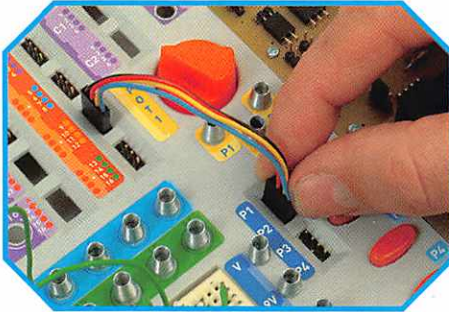
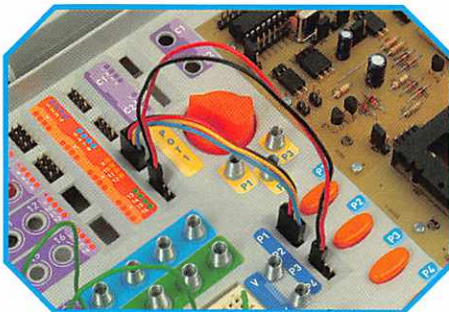
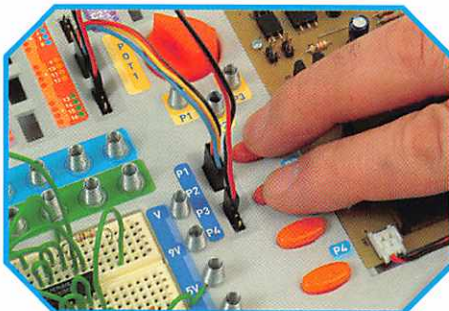
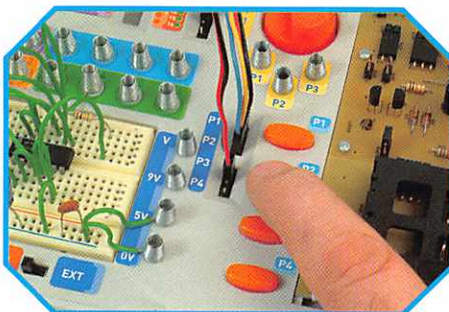
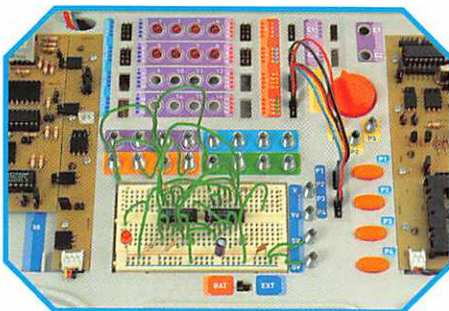
La soluzione

A partire da questo enunciato si ottiene la seguente tabella della verità:

P3	P2	P1	LEDA
1	0	0	1
0	1	1	1
0	0	1	1
1	0	1	1

Componenti sulla scheda Bread Board.

Collegamenti della scheda Bread Board.

Schema del circuito combinazionale.

LISTA DEI COMPONENTI

U1	Circuito integrato 4001
U2	Circuito integrato 4093
R1, R2	Resistenza 100 K (marrone, nero, giallo)
R3	Resistenza 330 K (arancio, arancio, giallo)
R4	Resistenza 1K8 (marrone, grigio, rosso)
C1	Condensatore 10 µF elettrolitico
C2	Condensatore 22 nF
LED A	Diodo LED rosso 5 mm.

Collegamento dei pulsanti P1 e P2.

Collegamento del pulsante P3.

Si illumina con la combinazione P1 = 1, P2 = 1, e P3 = 0.

Premendo solamente P2 non si attiva il LED.

Esperimento completato.

Da questa tabella della verità si ottiene l'equazione:

$$P3\,\overline{P2}\,\overline{P1} + \overline{P3}\,P2\,P1 + \overline{P3}\,\overline{P2}\,P1 + P3\,\overline{P2}\,P1 = F$$

Questa espressione si può semplificare fino ad arrivare a:

$$\overline{P3}\,P1 + P3\,\overline{P2}$$

Il modo intuitivo di verificare questa equazione è iniziare nello schema dalla sua parte sinistra, con la variabile, e proseguire annotando sullo schema stesso i cambiamenti e le espressioni all'uscita di ogni porta. Per ottenere la semplificazione della formula, è necessario ricordare che la somma di una variabile con il suo valore negato, è l'unità.

Il circuito

Vediamo lo schema per spiegare il circuito: la parte sinistra sono quelli è occupata dai tre pulsanti, ognuno di essi ha due terminali, dei quali uno è collegato al positivo dell'alimentazione e l'altro a uno o due ingressi delle porte logiche. In ognuno degli ingressi troviamo collegata una resistenza di pull-down la quale garantisce che gli ingressi così collegati siano 0 con i pulsanti in riposo. Come si può verificare, la combinazione e il collegamento delle porte, sono quelli necessari per soddisfare la funzione dell'enunciato. Ricordate che le combinazioni non rappresentate nella tabella, hanno come uscita lo 0 logico.

Montaggio

Il montaggio è piuttosto laborioso e necessita di un elevato numero di fili di collegamento, quindi deve essere realizzato con pazienza. Vi consigliamo di utilizzare il filo verde e di preparare alcuni pezzi da 5 cm con gli estremi spelati della parte isolante. L'alimentazione può essere da 5 o 9 V, e i condensatori C1 e C2 hanno il compito di filtrarla.

L'esperimento

Dopo aver alimentato il circuito è sufficiente azionare i pulsanti corrispondenti alle combinazioni delle variabili della tabella e in tutte queste combinazioni il risultato sarà 1, cioè si illuminerà il LED, per qualsiasi altra combinazione il LED non si illuminerà.

Esercizio 14: la memoria EEPROM, simulazione e pratica

Abbiamo sviluppato il codice che risolve l'applicazione dell'esercizio 14 di utilizzo della memoria EEPROM. Abbiamo anche compilato il codice alla ricerca di eventuali errori, e per ottenere il file in codice macchina da scrivere sul PIC. Non abbiamo ancora simulato il funzionamento del programma né realizzato il montaggio. Ora lavoreremo a queste fasi del progetto.

Simulazione

Dato che per la compilazione abbiamo dovuto creare un progetto associato a questo nostro codice, per simularlo dovremo semplicemente aprire il progetto creato e il file in assembler. Facendo partire MPLAB potremo aprire l'ultimo progetto con cui abbiamo lavorato, mediante una finestra come quella riportata nella figura. Nel caso corrisponda all'esercizio 14 selezioneremo Yes, in tutti gli altri casi selezioneremo No, e continueremo come spiegato precedentemente.

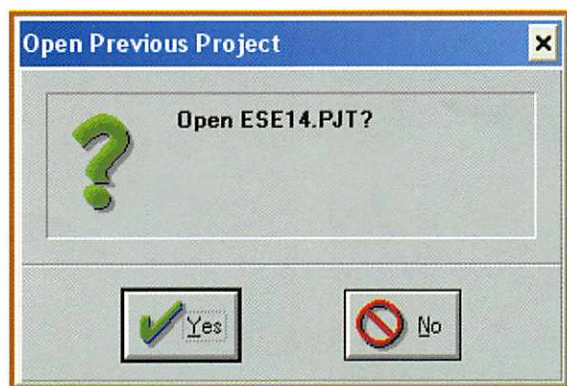
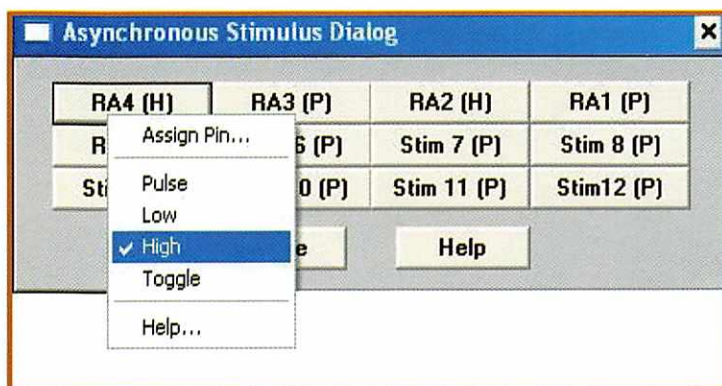
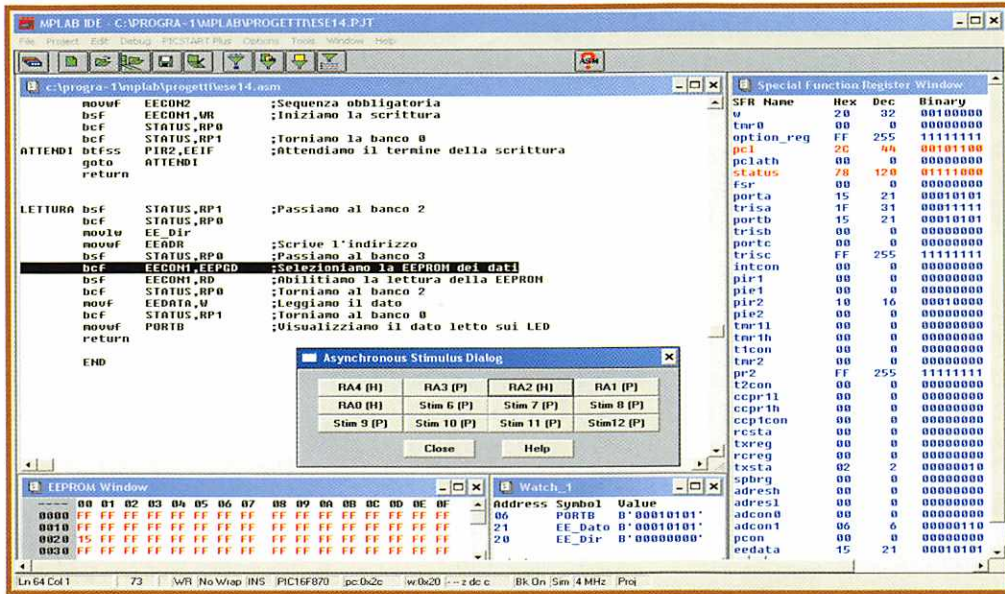
Con il progetto aperto e il codice visualizzato sul monitor, dobbiamo aprire le finestre di simulazione. Sceglieremo la finestra dei Registri delle Funzioni Speciali, la finestra della memoria EEPROM e la finestra dove visualizzare i registri più interessanti per la simulazione (visualizzeremo in binario PORTB, EE_Dato e EE_Dir).

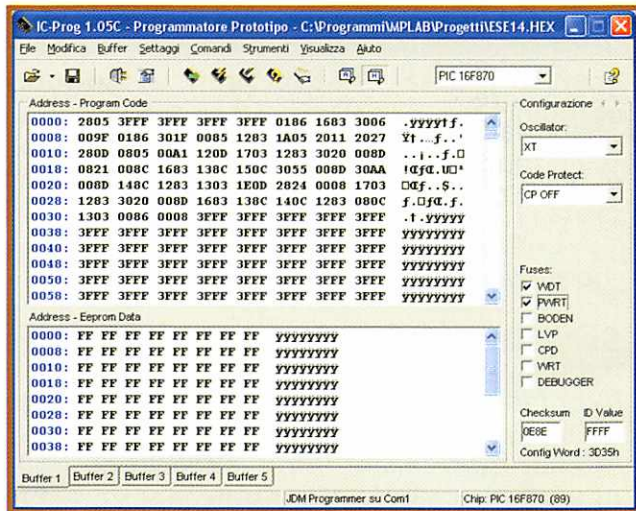
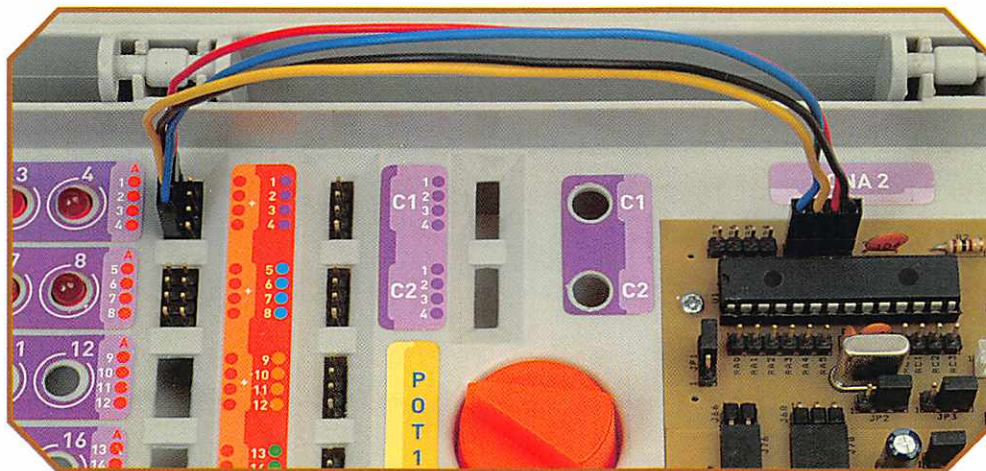
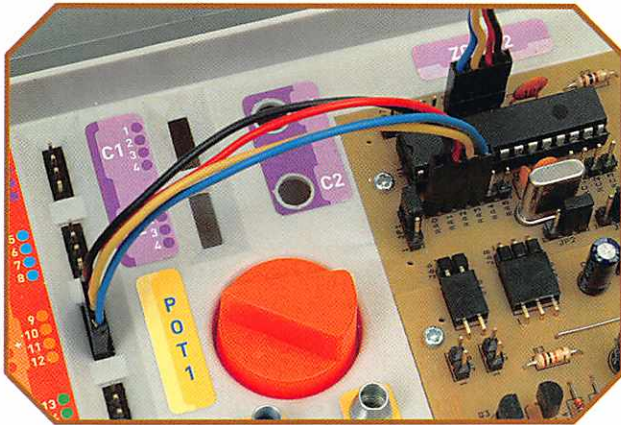
Per simulare gli ingressi dovremo utilizzare il simulatore di stimoli asincroni. Nel primo stimolo inseriremo RA4, dato che è l'ingresso che determina la scrittura di un valore nella memoria, continueremo con RA3, RA2, RA1 e RA0. Possiamo porre alcuni degli ingressi (RA3:RA0) a livello alto (High) prima di iniziare la simulazione passo a passo, ricordate però che per far sì che questi valori siano acquisiti, è necessario cliccare con il mouse su di essi uno alla volta in base a cosa si sta simulando.

Inizieremo la simulazione passo a passo premendo F7. Se la completiamo una volta vedremo che la porta B assumerà il valore '11111111', poiché legge un valore nella memoria. Se prima della volta successiva si clicca su RA4 passandolo a High, vedremo che il programma entrerà nella subroutine di Scrittura, passando alla EEPROM il valore dell'ingresso. Nella subroutine di Lettura si prenderà il valore della EEPROM e lo si porterà sulla porta di uscita.

A questo punto la simulazione è avvenuta con successo.

Scrittura del PIC

Ricordate che per la scrittura del PIC è necessario predisporre il laboratorio collegando il cavo di trasferimento, inserendo i ponticelli sui connettori JP8 e JP9 e sui connettori JP1, JP2 e JP3 sulle posizione 1 e 2. Faremo partire

MPLAB ci permette di aprire l'ultimo progetto con cui abbiamo lavorato.

Mediante il simulatore di stimoli asincroni, stimoliamo gli ingressi.

Aspetto di MPLAB durante la simulazione.

Prepariamo IC-Prog per la scrittura del PIC.

Se utilizziamo la matrice dei LED dobbiamo inserire i ponticelli sui catodi.

Uniamo gli anodi con la porta B.

Colleghiamo il segnale ai terminali di ingresso.

Montaggio dei segnali di ingresso sulla scheda Bread Board.

IC-Prog ed eseguiremo le fasi precedenti alla scrittura (lettura, cancellazione e verifica). Apriremo il file che vogliamo trasferire sul PIC e configureremo IC-Prog in modo che il trasferimento si realizzi correttamente, cambieremo il tipo di oscillatore a XT, verificheremo che la protezione del codice sia CP OFF e cliccheremo sui bit WDT e PWRT che fanno parte della parola di configurazione. Fatto questo potremmo continuare con la scrittura selezionando l'opzione Programma Tutto. Terminata la scrittura verificheremo il progetto appena eseguito con una lettura del contenuto del PIC.

Montaggio delle uscite

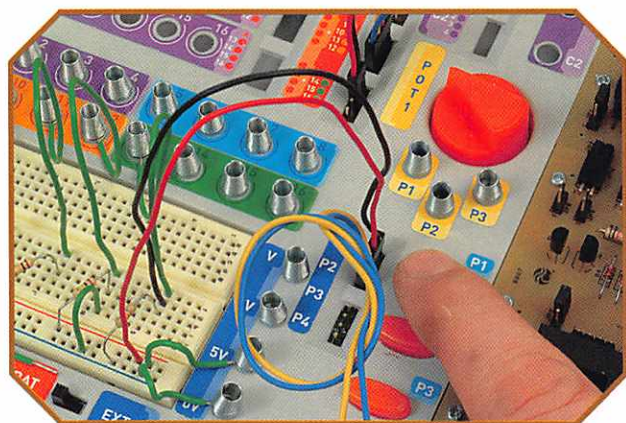
Prima di iniziare il montaggio dell'applicazione dobbiamo configurare nuovamente il laboratorio per riportarlo nella condizione precedente alla scrittura. I ponticelli dei connettori JP8 e JP9 devono essere tolti, quelli dei connettori JP1, JP2 e JP3 devono essere sulle posizioni 2 e 3, ed è necessario togliere il cavo di trasferimento.

Il valore dell'ingresso memorizzato sulla memoria EEPROM esce tramite la porta B visualizzandosi sulla matrice dei diodi LED. Dato che ciò che ci interessa vedere sono i valori corrispondenti agli ingressi RA3:RA0, RA4 lo utilizzeremo solamente per selezionare il momento della scrittura, utilizzeremo solamente i terminali di uscita RB3:RB0. Imposteremo i ponticelli sui catodi della prima fila dei diodi della matrice come si può vedere nella figura, utilizzeremo solamente questa fila di LED, e dato che vogliamo vedere quattro segnali lo faremo mediante un unico cavetto che collegherà gli anodi con i terminali RB3:RB0 della porta B.

Colleghiamo il terminale RA4.

Montaggio degli ingressi

Ingressi RA3:RA0

Per simulare sul laboratorio gli interruttori collegati a questi ingressi eseguiremo un montaggio sulla scheda Bread Board. I segnali di questa scheda li porteremo alla porta utilizzando le molle di collegamento, quindi uniremo mediante un cavetto i terminale RA3:RA0, con il connettore volante che prende i segnali dalle molle con etichetta arancio (dal 9 al 12). Sulla scheda Bread Board realizzeremo un montaggio molto semplice, mediante delle resistenze. I terminali RA3:RA0,

Tramite il pulsante determineremo quando far passare il segnale di ingresso sull'uscita.

che sarebbe come dire le molle dalla 9 alla 12, si collegano direttamente al positivo tramite una resistenza.

Per cambiare lo stato di questi segnali prepareremo dei ponticelli che potremo collegare al negativo, se vogliamo uno 0 sull'ingresso, o non collegarli se vogliamo un 1. Nella figura possiamo vedere il montaggio che abbiamo appena spiegato.

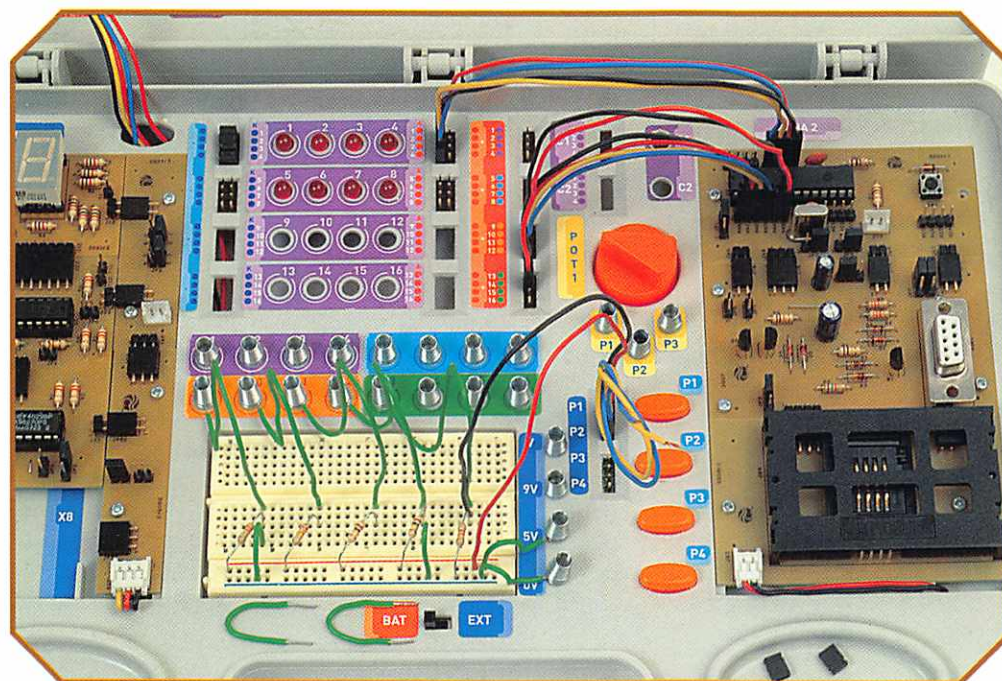
Ingresso RA4

Differenziamo questo ingresso dagli altri, perché sarà quello che determina in quale momento i segnali presenti sugli altri terminali di ingresso saranno scritti sulla memoria EEPROM. Per far arrivare questo segnale al terminale corrispondente della porta A (RA4) utilizzeremo le molle di collegamento, quindi potremo unire mediante un cavetto, il connettore che corrisponde al terminale RA4. Nella figura della pagina precedente possiamo vedere questo collegamento.

Alla molla, ovvero allo stesso terminale RA4, collegheremo una resistenza che dall'altro capo andrà al negativo (0V). Invece di utilizzare un ponticello come per i segnali precedenti, potremo attivare il segnale mediante il pulsante P1, che verrà inserito fra la molla e i 5 V. Nella figura si può verificare come viene collegato il pulsante alla scheda Bread Board mediante un cavetto.

Prova di funzionamento

Per provare il corretto funzionamento impostate alcuni degli ingressi RA3:RA0 al positivo (senza il ponticello) e premete il pulsante P1, vedrete come verrà riportato sull'uscita il valore presente sull'ingresso.

Provate ora a cambiare gli ingressi e verificate che l'uscita non cambia. L'uscita cambia solamente quando si attiva RA4, cioè quando mediante il pulsante passeremo il nuovo valore sulla memoria EEPROM. Se non si attiva P1, il valore visualizzato sarà il valore che il PIC contiene nella memoria EEPROM.

Montaggio dell'applicazione completato.

Esercizio 16: l'importanza di SLEEP e del WDT, il programma

Facciamo un piccolo salto negli esercizi per presentarne uno molto semplice che illustra in modo eccezionale l'importanza del Watch-Dog Timer (WDT) combinato con l'istruzione Sleep. Questo esercizio si trova risolto nel secondo CD allegato all'opera, con il nome "ese16.asm".

Enunciato

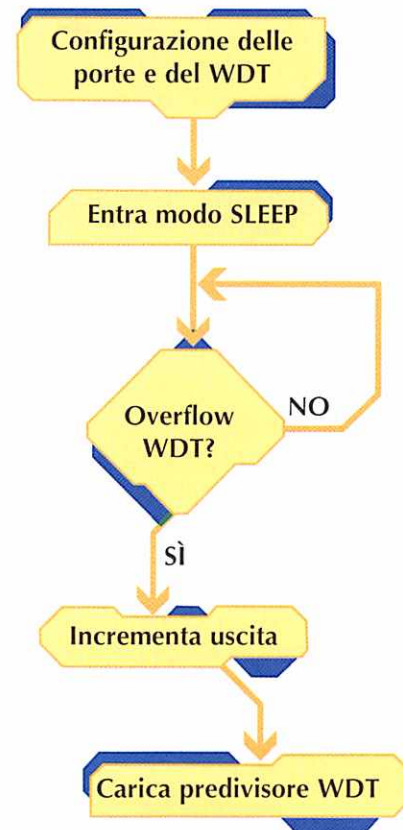
Mediante l'istruzione SLEEP vogliamo impostare il PIC in modo standby di basso consumo, dal quale uscirà ogni volta che il WDT va in overflow. In questo momento si genera un incremento del valore della porta B che funzionerà come contatore binario e nuovamente tornerà nella situazione standbay.

Il predivisore che si associa al WDT sarà compreso fra 1 e 128, in base allo stato logico degli interruttori RA0-RA2. Il valore nominale del WDT è di 18 ms. Cioè, con un predivisore di 1, il PIC si risveglierà ogni 18 ms, con un predivisore di 128 lo farà ogni 2,3 secondi.

Organigramma

L'enunciato specifica chiaramente ciò che si vuole dal programma, quindi risulta molto semplice predisporre una soluzione generale dell'esercizio. Nell'immagine si può vedere l'organigramma che corrisponde all'enunciato.

Codice

Iniziamo, come d'abitudine, con i commenti che spiegano il funzionamento del program-

Organigramma dell'applicazione.

```
;ESERCIZIO: Il modo "sleep" e il "wake-up" (risveglio) mediante il watch-dog Timer (WDT)
;--------------------------------------------------------------------------------
;
;Questo esempio vuole illustrare l'utilizzo dell'istruzione SLEEP per impostare il PIC in
;modo standby a basso consumo. La riattivazione del medesimo avviene ogni volta che il WDT
;va in overflow. In quel momento si produce un incremento di valore della porta B che funziona
;come un contatore binario dopodiché il PIC tornerà nuovamente alla situazione di standby.
;
;Il prescaler si associerà al WDT e sarà compreso fra 1 e 128, in base allo stato
;logico degli interruttori RA0-RA2.
;
;Il valore nominale del WDT è di 18 ms. Quindi con un prescaler di 1, il PIC si "risveglierà"
;ogni 18 ms, con un prescaler da 128, lo farà ogni 2,3 secondi.

                List    p=16F870            ;Tipo di processore
                include "P16F870.INC"       ;Definizione dei registri interni

                org     0x00                ;Vector di Reset
                goto    Inizio
                org     0x05                ;Salva il vector di interrupt
```

Inizieremo tutti i programmi con dei commenti, le definizioni del PIC e le direttive di organizzazione.

Dispositivi	Registri da configurare
Porta A	TRISA, ADCON1
Porta B	TRISB
Watch-Dog timer	OPTION_REG

Dispositivi con cui lavoreremo.

ma. Per questi esercizi, vi consigliamo di copiare l'enunciato così com'è all'interno del file di testo in cui sviluppiamo il nostro codice. Questi non saranno gli unici commenti all'interno del programma, dato che un buon progettista utilizza i commenti praticamente su tutte le linee di codice, per spiegarne in dettaglio il funzionamento.

Definiamo il microcontroller e la libreria che contiene la definizione dei registri e mediante le direttive ORG organizziamo all'interno della memoria il codice che scriveremo.

La prima parte del codice, a partire dall'etichetta Inizio, contiene la configurazione dei dispositivi che vogliamo utilizzare. Impostiamo a 0 tutti i bit del registro TRISB; per configurare la porta B come uscita, configureremo la porta A come ingresso, impostando a 1 i cinque bit meno significativi del registro TRISA, configureremo gli ingressi come digitali mediante il registro ADCON1 e mediante il registro OPTION_REG configureremo il WDT.

Modo SLEEP o di basso consumo

Inizieremo il ciclo impostando un'etichetta a cui poterci indirizzare al termine di questo blocco di programmazione. La prima istruzione del ciclo sarà l'istruzione SLEEP. Questa istruzione manda il microcontroller in uno stato di riposo o di basso consumo. Si disattivano parte dei dispositivi in modo che il PIC rimanga con il minimo necessario per potersi risvegliare. Si tratta di uno stato simile a quello di ibernazione o sospensione in cui entra un PC e dal quale esce solamente se si produce un evento esterno. Nel PIC, per uscire dal modo riposo, in altre parole per risvegliarlo, possiamo utilizzare il WDT, oppure un reset o un interrupt esterno. In molte applicazioni non è necessario che il PIC funzioni costantemente poiché rimane in attesa di un evento esterno (come ad esempio l'inserimento di un codice per accedere a una zona determinata). In queste applicazioni è molto importante utilizzare correttamente il modo SLEEP.

PS2:PS0	Range del WDT
000	1:1
001	1:2
010	1:4
011	1:8
100	1:16
101	1:32
110	1:64
111	1:128

Range del predivisore del WDT in funzione dei bit OPTION_REG <2:0>.

Programmazione del ciclo di SLEEP

Dopo aver configurato tutti i dispositivi, creiamo un ciclo assegnandogli un'etichetta a cui poter saltare in seguito. Il ciclo inizia con l'istruzione SLEEP, con la quale mandiamo il PIC in modo di basso consumo; in questo modo il microocontroller non fa nulla, si trova addormentato, e uscirà dal suo letargo solamente quando il WDT va in overflow. In questo mo-

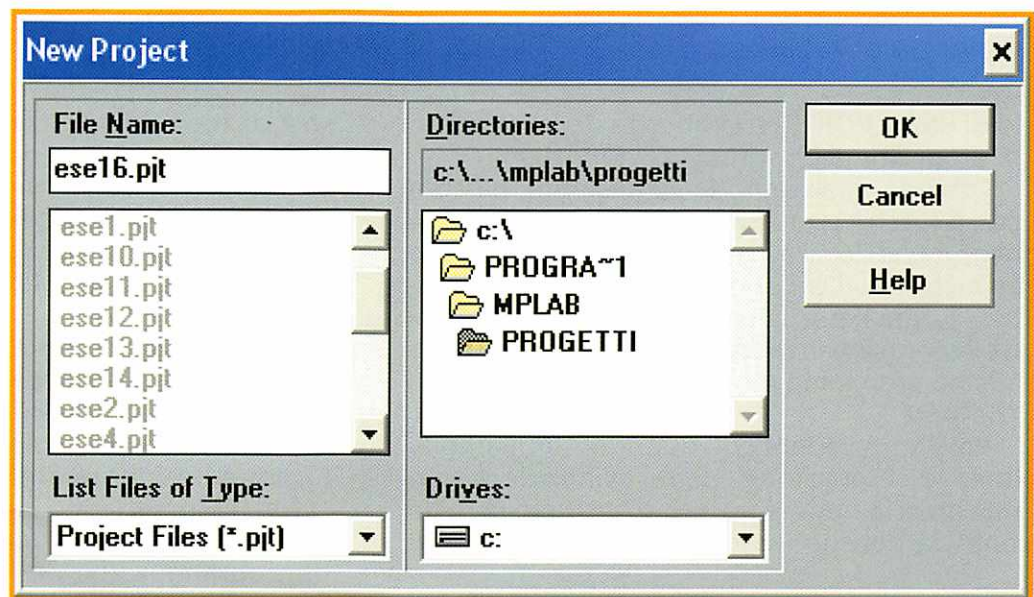
```
Inizio          clrf    PORTB           ;Cancella i latch di uscita
                bsf     STATUS,RP0      ;Seleziona il banco 1
                clrf    TRISB           ;Porta B si configura come uscita
                movlw   b'00011111'
                movwf   TRISA           ;RA0-RA4 ingressi
                movlw   b'00000110'
                movwf   ADCON1
                movlw   b'00001000'
                movwf   OPTION_REG      ;Prescaler da 1 per il WDT
                bcf     STATUS,RP0      ;Seleziona il banco 0
```

Codice di configurazione dei dispositivi.

```
Loop        sleep                       ;Modo Standby

            incf    PORTB,F             ;Incrementa il contatore binario sulla porta B
            movf    PORTA,W
            andlw   b'00000111'         ;Legge lo stato degli interruttori RA0-RA2
            iorlw   b'00001000'
            bsf     STATUS,RP0          ;Seleziona il banco 1
            movwf   OPTION_REG          ;Regola il valore del prescaler
            bcf     STATUS,RP0          ;Seleziona il banco 1
            goto    Loop                ;Torna al modo Standby

            end                         ;Fine del programma sorgente
```

Codice del ciclo di SLEEP.

Selezionando New Project creiamo un nuovo progetto.

mento dobbiamo incrementare il valore da visualizzare tramite i LED collegati alla porta B, che non è altro che il numero di volte che il PIC è uscito dal modo riposo. Per preparare un nuovo SLEEP prendiamo il valore del predivisore del WDT, che viene caricato sulla porta A e lo scriviamo sul registro OPTION_REG, saltando poi all'inizio del ciclo per ripetere tutto il processo.

Compilazione

Abbiamo terminato quello che pensiamo sia il codice dell'applicazione, però per accertarci che quello che abbiamo progettato risponda all'enunciato, dobbiamo passare alla fase di compilazione, simulazione e montaggio in modo soddisfacente. Faremo partire MPLAB e creeremo un nuovo progetto. Ricordate che è conveniente utilizzare lo stesso nome sia per il progetto che per il codice, o dare un nome al progetto che indichi in modo esplicito ciò che esso sviluppa. Nel nostro caso possiamo chiamare il progetto "ese16.pjt" o "sleep.pjt". Creando un nuovo progetto si apre la finestra di edizione di quest'ultimo e tramite questa finestra assoceremo il file che contiene il codice.

Apriamo il file per visualizzare il codice sul monitor e compiliamo il progetto selezionando l'opzione Build All. Se la compilazione avviene correttamente, apparirà la videata riportata nella pagina successiva in cui veniamo informati che la compilazione è stata eseguita con successo (Build completed successfully).

Simulazione

Per simulare questa applicazione apriremo le finestre dei Registri delle Funzioni Speciali, una finestra con i registri più importanti (nel nostro caso sarà sufficiente visualizzare la por-

```
Build Results
Building ESE16.HEX...

Compiling ESE16.ASM:
Command line: "D:\PROGRA~1\MPLAB\MPASMWIN.EXE /p16F870 /q C:\PROGRA~1\MPLAB\PROGETTI\ESE16.ASM"
Message[302] C:\PROGRA~1\MPLAB\PROGETTI\ESE16.ASM 25 : Register in operand not in bank 0.  Ensure
Message[302] C:\PROGRA~1\MPLAB\PROGETTI\ESE16.ASM 27 : Register in operand not in bank 0.  Ensure
Message[302] C:\PROGRA~1\MPLAB\PROGETTI\ESE16.ASM 29 : Register in operand not in bank 0.  Ensure
Message[302] C:\PROGRA~1\MPLAB\PROGETTI\ESE16.ASM 31 : Register in operand not in bank 0.  Ensure
Message[302] C:\PROGRA~1\MPLAB\PROGETTI\ESE16.ASM 42 : Register in operand not in bank 0.  Ensure

Build completed successfully.
```

Risultato della compilazione.

ta di uscita PORTB) e la finestra della memoria di programma (Program Memory).

Quando inizieremo la simulazione passo a passo (premeremo F7 e simuleremo linea a linea) il codice si eseguirà correttamente fino ad arrivare all'istruzione SLEEP. A questo punto si ferma l'esecuzione, dato che il microcontroller entrerà in modo riposo, infatti anche se adesso premiamo molte volte F7 continuerà a "dormire".

Possiamo vedere che quando simuliamo la linea di esecuzione viene evidenziata con una striscia nera, e questo si verifica sia nella finestra dei file in assembler che nella finestra di memoria del programma. Per uscire dallo SLEEP dopo esservi entrati con la simulazione, faremo un doppio clic con il pulsante sinistro del mouse sulla linea 17 della memoria di programma, la linea che si trova evidenziata col colore rosso e la cui istruzione è "incf 0x6".

Con questo doppio clic, la linea di stato di MPLAB passerà a colore giallo (linea inferiore della finestra), per alcuni secondi. Automaticamente sparisce il colore giallo e premendo F7 potremo continuare con la simulazione.

Nel caso rimanessimo bloccati, potremo cliccare con il mouse sulla linea della memoria di programma che desideriamo, per fare un salto nella simulazione.

La porta B si incrementa correttamente ogni volta che usciamo dallo SLEEP, quindi la simulazione può essere considerata corretta e terminata.

```
Program Memory Window
1          0000  2805          goto   Inizio
2          0001  3FFF          addlw  0xFF
3          0002  3FFF          addlw  0xFF
4          0003  3FFF          addlw  0xFF
5          0004  3FFF          addlw  0xFF
6          0005  0186  Inizio  clrf   0x6
7          0006  1683          bsf    0x3,0x5
8          0007  0186          clrf   0x6
9          0008  301F          movlw  0x1F
10         0009  0085          movwf  0x5
11         000A  3006          movlw  0x6
12         000B  009F          movwf  0x1F
13         000C  3008          movlw  0x8
14         000D  0081          movwf  0x1
15         000E  1283          bcf    0x3,0x5
16         000F  0063  Loop    sleep
17    B... 0010  0A86          incf   0x6
18    B... 0011  0805          movf   0x5,W
19         0012  3907          andlw  0x7
20    B... 0013  3808          iorlw  0x8
21         0014  1683          bsf    0x3,0x5
22    B... 0015  0081          movwf  0x1
23         0016  1283          bcf    0x3,0x5
24         0017  280F          goto   Loop
25         0018  3FFF          addlw  0xFF
26         0019  3FFF          addlw  0xFF
27         001A  3FFF          addlw  0xFF
28         001B  3FFF          addlw  0xFF
29         001C  3FFF          addlw  0xFF
30         001D  3FFF          addlw  0xFF
```

Finestra della memoria di programma.

La linea di stato cambia nel colore giallo.